CLAIMS

What is claimed is:

1. A method, comprising:

determining a new enqueue slot of a circular queue having N slots into which a queue element may be enqueued;

determining whether the circular queue is full via executing a check comparing relative positions of the new enqueue slot and a current dequeue slot ("CDS"); and

enqueuing the queue element into the new enqueue slot, if the circular queue is not full.

2. The method of claim 1 wherein determining whether the circular queue is full via executing the check comprises determining whether enqueuing the queue element into the new enqueue slot would result in an overflow condition of the circular queue via executing the check.

3. The method of claim 2, further comprising:

setting a last enqueue slot ("LES") pointer currently designating an old enqueue slot to designate the new enqueue slot after determining the new enqueue slot;

dropping the enqueue element, if the overflow condition would result from enqueuing the queue element into the new enqueue slot; and

resetting the LES pointer to designate the old enqueue slot, if the overflow condition would result from enqueuing the queue element.

4. The method of claim 3, wherein executing the check further comprises determining whether the following relation is true:

$$((CDS^{N} - LES^{N}) \bmod N) < M ,$$

wherein CDS^{N} represents CDS mod N, LES^{N} represents LES mod N, and M represents a number less than N.

5. The method of claim 4 wherein M is equal to or greater than a maximum number of slots that may be enqueued with queue elements during a delay period for updating a dequeue counter.

6. The method of claim 1 wherein determining the new enqueue slot of the circular queue into which the queue element may be enqueued comprises determining the new enqueue slot according to a pre-sort deficit round robin enqueuing scheme.

7. The method of claim 2 wherein each of the N slots of the circular queue can buffer multiple queue elements corresponding to multiple logical queues, wherein the queue element corresponds to a particular one of the multiple logical queues, and wherein the new enqueue slot corresponds to the particular one of the multiple logical queues.

8. The method of claim 7 wherein determining whether the circular queue is full comprises determining whether enqueuing the queue element into the new enqueue slot

of the circular queue would result in an overflow condition of the particular one of the multiple logical queues.

9. A method, comprising:

dequeuing a queue element from a current dequeue slot ("CDS") of a circular queue having N slots;

designating a new CDS;

determining whether the circular queue is empty via executing a first check comparing relative positions of the new CDS and a last enqueued slot ("LES"); and

setting the LES to the new CDS, if the circular queue is determined to be empty.

10. The method of claim 9 wherein the CDS is designated by a CDS pointer, wherein designating the new CDS comprises incrementing the CDS pointer to designate the new CDS, wherein the LES is designated by a LES pointer, and wherein setting the LES to the new CDS comprises setting the LES pointer to designate the new CDS, if the circular queue is determined to be empty.

11. The method of claim 9 wherein determining whether the circular queue is empty further comprises executing a second check prior to executing the first check, the second check comprising:

determining whether an enqueue count is equal to a dequeue count.

12. The method of claim 11, further comprising incrementing the dequeue count after dequeuing the queue element from the CDS of the circular queue.

13. The method of claim 9 wherein executing the first check further comprises determining whether the following relation is true:

$$((CDS^N - LES^N) \bmod N) < M$$

wherein CDS^N represents CDS mod N, LES^N represents LES mod N, and M represents a number less than N.

14. The method of claim 9 wherein M is equal to or greater than a maximum number of slots of the circular queue that may be enqueued with queue elements during a delay period for updating a dequeue counter.

15. The method of claim 9 wherein each of the N slots of the circular queue can buffer multiple queue elements corresponding to multiple logical queues, wherein the queue element corresponds to a particular one of the multiple logical queues, and wherein LES corresponds to the particular one of the multiple logical queues.

16. The method of claim 15 wherein determining whether the circular queue is empty via executing the first check comprises determining whether the particular one of the multiple logical queues is empty via executing the first check.

17. A machine-accessible medium that provides instructions that, if executed by a machine, will cause the machine to perform operations comprising:

dequeuing a first queue element from a current dequeue slot ("CDS") of a circular queue having N slots, the CDS designated by a CDS pointer;

incrementing the CDS pointer to designate a new CDS; and

determining whether the circular queue is empty after the incrementing via executing a first check comparing relative positions within the circular queue designated by the CDS pointer and a last enqueued slot ("LES") pointer.

18. The machine-accessible medium of claim 17, further providing instructions that, if executed by the machine, will cause the machine to perform further operations, comprising:

setting the LES pointer currently designating an old enqueue slot of the circular queue to designate a new enqueue slot of the circular queue into which a second queue element may be enqueued;

determining whether enqueuing the second queue element into the new enqueue slot would result in an overflow condition of the circular queue via re-executing the first check after setting the LES pointer to designate the new enqueue slot.

19. The machine-accessible medium of claim 18, further providing instructions that, if executed by the machine, will cause the machine to perform further operations, comprising:

enqueuing the second queue element into the new enqueue slot, if the overflow condition would not result from enqueuing the second queue element into the new enqueue slot;

dropping the second enqueue element, if the overflow condition would result from enqueuing the second queue element into the new enqueue slot; and

resetting the LES pointer to designate the old enqueue slot, if the overflow condition would result from enqueuing the second queue element into the new enqueue slot.

20. The machine-accessible medium of claim 18, further providing instructions that, if executed by the machine, will cause the machine to perform a further operation, comprising determining the new enqueue slot of the circular queue into which the second queue element may be enqueued.

21. The machine-accessible medium of claim 20, wherein determining the new enqueue slot of the circular queue into which the second queue element may be enqueued comprises determining the new enqueue slot according to a per-sort deficit round robin queuing scheme.

22. The machine-accessible medium of claim 17, further providing instructions that, if executed by the machine, will cause the machine to perform further operations, comprising:

setting the LES pointer to designate the current dequeue slot, if the circular queue is determined to be empty.

23. The machine-accessible medium of claim 22 wherein the determining whether the circular queue is empty further comprises executing a second check prior to executing the first check, the second check comprising:

determining whether an enqueue count is equal to a dequeue count, wherein the enqueue count is incremented each time a queue element is enqueued and the dequeue count is incremented each time a queue element is dequeued.

24. The machine-accessible medium of claim 18 wherein executing and re-executing the first check comparing the relative positions within the circular queue designated by the CDS pointer and the LES pointer comprises determining whether the following relation is true:

$$((CDS^{N} - LES^{N}) \bmod N) < M$$

wherein CDS^{N} represents CDS mod N, LES^{N} represents LES mod N, and M represents a number less than N.

25. The machine-accessible medium of claim 18 wherein each of the N slots of the circular queue can buffer multiple queue elements corresponding to multiple logical queues, and wherein the first queue element, the second queue element, and the LES pointer correspond to a particular one of the multiple logical queues.

26. The machine-accessible medium of claim 25 wherein determining whether the circular queue is empty comprises determining whether the particular one of the logical queues is empty and wherein determining whether enqueuing the second queue element would result in an overflow condition of the circular queue comprises determining whether enqueuing the second queue element would result in an overflow condition of the particular one of the logical queues.

27. A router system, comprising:

an input port to receive a first data unit from a first network link;

a circular queue having N slots to queue a first queue element;

a network processor communicatively coupled to the input port and the circular queue, the network processor coupled to:

set a last enqueue slot ("LES") pointer to designate a new enqueue slot of the N slots into which the first queue element may be enqueued; and

determine whether enqueuing the first queue element into the new enqueue slot would result in an overflow condition of the circular queue via executing a first check after setting the LES pointer, the first check comparing relative positions within the circular queue designated by the LES pointer and a current dequeue pointer ("CDS"); and

an output port communicatively coupled to the network processor to transmit the first data unit to a second network link in response to dequeuing the first queue element from the circular queue.

28. The router system of claim 27 wherein the network processor is further coupled to:

enqueue the first queue element into the new enqueue slot, if the overflow condition would not result from the enqueuing the first queue element into the new enqueue slot;

drop the first data unit, if the overflow condition would result from enqueuing the first queue element into the new enqueue slot; and

reset the LES pointer to designate a previous enqueue slot, if the overflow condition would result from enqueuing the first queue element into the new enqueue slot.

29. The router system of claim 28 wherein the network processor is further coupled to:

dequeue a second queue element from a CDS of the N slots designated by the CDS pointer;

increment the CDS pointer to designate a new CDS; and

determine whether the circular queue is empty via re-executing the first check after the increment of the CDS pointer.

30. The router system of claim 28 wherein the network processor is further to determine whether the circular queue is empty via executing a second check prior to re-executing the first check, the second check comprising:

determining whether an enqueue count is equal to a dequeue count, wherein the network processor is to increment the enqueue count each time a queue element is enqueued and to increment the dequeue count each time a queue element is dequeued.

31. The router system of claim 27 wherein executing the first check comparing the relative positions within the circular queue designated by the CDS pointer and the LES pointer comprises determining whether the following relation is true:

$$((CDS^N - LES^N)\,\mathrm{mod}\,N) < M$$

wherein CDS^N represents CDS mod N, LES^N represents LES mod N, and M represents a number less than N.

32. The router system of claim 27 wherein the first data unit comprises one of a packet, a cell, and a frame.

33. The router system of claim 27 wherein the first queue element comprise a pointer to a memory location containing the data unit.

34. The router system of claim 29 wherein each of the N slots of the circular queue can buffer multiple queue elements corresponding to multiple logical queues and wherein the first queue element, the second queue element, and the LES pointer correspond to a particular one of the multiple logical queues.

35. The router system of claim 34 wherein determining whether the circular queue is empty comprises determining whether the particular one of the logical queues is empty and wherein determining whether enqueuing the first queue element would result in an overflow condition of the circular queue comprises determining whether enqueuing the first queue element would result in an overflow condition of the particular one of the logical queues.